PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 12/56 | A1

(11) International Publication Number: **WO 99/66678**

(43) International Publication Date: 23 December 1999 (23.12.99)

(21) International Application Number: PCT/EP99/04377

(22) International Filing Date: 15 June 1999 (15.06.99)

(30) Priority Data:
09/097,898 16 June 1998 (16.06.98) US

(71) Applicant: ALCATEL [FR/FR]; 54, rue la Boétie, F–75008 Paris (FR).

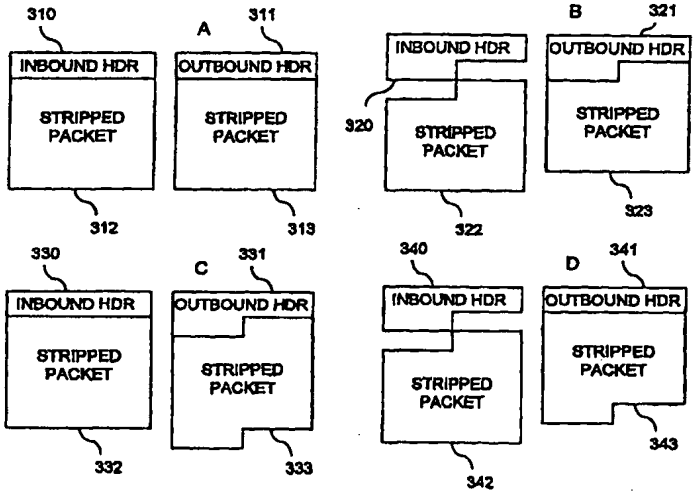
(72) Inventor: BERGENFELD, Bruce, Eric; 5420 Mark Court, Agoura Hills, CA 91301 (US).

(74) Agents: RAUSCH, Gabriele et al.; Alcatel, Intellectual Property Dept. Stg, Postfach 300 929, D–70449 Stuttgart (DE).

(81) Designated States: CN, JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: PACKET ASSEMBLY HARDWARE FOR DATA COMMUNICATION SWITCH



(57) Abstract

An "on the fly" packet assembly for a data communication switching engine assembles headers and stripped packets which are separately–sourced in multi–bit bursts while correcting any misalignment created by such "chunky" transfer. When an inbound header and corresponding outbound header initially have a half–width divergence (i.e., one ends on a full–width and the other ends on a half–width), an alignment unit realigns the stripped packet by a half–width to align the last half–width of the outbound header and the first half–width of the stripped packet. Also, when an outbound header ends on a half–width, a merger multiplexor combines the last half–width of the outbound header and the first half–width of the stripped packet to bridge the gap which would otherwise remain between the outbound header and the stripped packet. The serially–implemented alignment and merger operations format the outbound header and stripped packet into an encapsulated packet which may be readily transferred in a contiguous manner on an output. An update unit may be implemented in the packet assembly to perform "on the fly" updates of selective fields in the outbound header and stripped packet.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

PACKET ASSEMBLY HARDWARE FOR DATA COMMUNICATION SWITCH

BACKGROUND OF THE INVENTION

The present invention relates to data communication switching and, more particularly, to data communication switching hardware for assembling packets for transmission on an output.

Data communication switches have switching engines which receive inbound packets on inputs and switch them as outbound packets on outputs. The inbound packets generally have an inbound header including media access control (MAC) addresses (Layer Two switching), network addresses (Layer Three switching) or transport layer identifiers (Layer Four switching). The inbound header is stripped from the packet and the addresses or identifiers contained therein are resolved to an outbound header suitable for interpretation at the packet's "next hop" through the network. The outbound header is appended to the stripped packet to form an "encapsulated" outbound packet.

To improve switching speed, the outbound header and the stripped packet are often separately-sourced into a packet assembly in a series of "bursts" of K-bit wide data, where K is greater than one. Such "chunky" packet transfer presents some technical obstacles to "on the fly" encapsulation. For one, if the inbound header ended on a whole width (i.e., both half-widths of the final burst containing data for the inbound header included valid header data) but the corresponding outbound header ends on a half-width (i.e., only one half-width of the final burst containing data for the outbound header includes valid header data), or vice versa, the start of the stripped packet will, if not advanced (or delayed) appropriately, trail (or lead) the outbound header by a half-width. For another, if the outbound header ends on a half-width, the stripped packet will, if not

advanced (or delayed) appropriately, trail (or lead) the outbound header by either a half-width or a whole width. If this misalignment created by "chunky" transfer from multiple sources is not corrected, packets will be encapsulated improperly and spurious data or system errors may result.

SUMMARY OF THE INVENTION

In its most basic feature, the present invention provides an "on the fly" packet assembly for a data communication switching engine which assembles packet elements separately-sourced in multi-bit bursts while correcting any misalignment created by "chunky" transfer. When an inbound header and corresponding outbound header have a half-width divergence (i.e., one ends on a whole width and the other ends on a half-width), the stripped packet is realigned by a half-width to align the last half-width of the outbound header and the first half-width of the stripped packet. Also, when an outbound header ends on a half-width, the last half-width of the outbound header and the first half-width of the stripped packet are combined to bridge the gap which would otherwise remain between the outbound header and the stripped packet. The serially-implemented alignment and merger operations format the outbound header and stripped packet into an encapsulated packet which may be transferred contiguously on an output.

The foregoing advantages may be achieved with the expedient of an alignment unit followed by a merger multiplexor. The alignment unit may be arranged as a selectively implemented half-width register followed by one or more full-width registers. When an inbound header ended on a whole width and the outbound header ends on a half-width, the half-width register captures half-widths of the stripped packet while the other half-widths are allowed to flow directly into a full-width register. This activity

creates an offset in the stripped packet which matches that of the outbound header (i.e., afterward the outbound header ends and the stripped packet begins on a half-width). Similarly, when an inbound header ended on a half-width and the outbound header ends on a whole width, the half-width register captures half-widths of the stripped packet while the other half-widths are allowed to flow directly into a full-width register. This activity creates an offset in the stripped packet which matches that of the outbound header (i.e., afterward the outbound header ends and the stripped begins on a whole burst). When an inbound header and outbound header do not initially have a half-width divergence, the half-width register is bypassed and whole widths of the stripped packet are allowed to flow directly into a full-width register.

The merger multiplexor may be arranged as a two-width to one-width multiplexor which from a given two-width input may select as an output either a whole width from the outbound header, a whole width from the stripped packet, or a half-width from each. More particularly, when the outbound header ends on a whole width, the merger multiplexor always selects whole widths from the outbound header until the entire header has been transferred and then selects whole bursts from the stripped packet until the entire stripped packet has been transferred. When the outbound header ends on a half-width, the same selection sequence is followed except when the last half-width of the outbound header and the first half-width of the stripped packet are made available as inputs. During that transition, the merger multiplexor selects the last half-width of the outbound header and the first half-width of the stripped packet to "merge" the output header with the stripped packet. Because of the previous alignment function performed selectively by the alignment unit, half-widths may always be clocked to arrive at the

merger multiplexor in the manner required for the merger multiplexor to combine the outbound header and the stripped packet into an encapsulated packet adapted for contiguous transfer.

In another aspect of the invention, an update unit may be interposed to selectively update half-width, full-width or multiple width fields in the packet.

These and other objects of the present invention may be better understood by reference to the following detailed description, taken in conjunction with the accompanying drawings which are briefly described below. Of course, the actual scope of the invention is defined by the appended claims.

BRIEF DESCRIPTION OF THE DRAWING

Figure 1 is a block diagram of a data communication switching engine;

Figure 2 is a more detailed block diagram of the packet assembly element of the switching engine of Figure 1;

Figures 3A through 3D illustrate the different inbound header/outbound header combinations which the packet assembly element of Figure 2 is arranged advantageously to treat differently to form encapsulated packets;

Figure 4 is a more detailed block diagram of the alignment unit of the switching engine of Figure 1;

Figure 5 is a more detailed block diagram of the update unit of the switching engine of Figure 1;

Figure 6 is a more detailed block diagram of the merger multiplexor of the switching engine of Figure 1 and its associated logic;

Figure 7A illustrates the serial operation of the alignment unit and the merger multiplexor in the situation where the inbound header and outbound header do not have a half-width divergence and the outbound header ends on a whole width;

Figure 7B illustrates the serial operation of the alignment unit and the merger multiplexor in the situation where the inbound header and outbound header do not have a half-width divergence and the outbound header ends on a half-width;

Figure 7C illustrates the serial operation of the alignment unit and the merger multiplexor in the situation where the inbound header and outbound header have a half-width divergence and the outbound header ends on a half-width; and

Figure 7D illustrates the serial operation of the alignment unit and the merger multiplexor in the situation where the inbound header and outbound header have a half-width divergence and the outbound header ends on a whole width.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

In Figure 1, a switching engine 100 in which the present invention may be implemented is shown. Inbound packets arrive in receive FIFO 110. Identifiers in the headers of inbound packets are transmitted to switching logic 120 for a switching decision. If the switching decision indicates forwarding, switching logic 120 transmits a forwarding index to header table 140 to retrieve an appropriate outbound header. Identifiers transmitted to switching logic 120 may include Open System Interconnection (OSI) Layer Two (Bridging) Layer Three (Network) and Layer Four (Transport) addresses and identifiers, by way of example. Switching logic 120 may make the switching decision by performing associative comparisons of such identifiers with known identifiers stored in a memory within switching logic 120. Such a memory may be a

content-addressable memory (CAM) or may be a random access memory (RAM). One example of a RAM-based implementation for switching logic 120 is described in Application Serial No. 08/964,597 entitled CUSTOM CIRCUITRY FOR ADAPTIVE HARDWARE ROUTING ENGINE, owned by the assignee hereof. Data in the inbound packet which will not be subject to overwrite by the outbound header are stored in packet FIFO 130 pending the results of the switching decision in switching logic 120. In this way, such data avoid being overwritten by the contents of other inbound packets arriving at receive FIFO 120. Packet assembly 150 receives bursts of data separately from header FIFO 145 and from packet FIFO 130 and combines such data "on the fly" into encapsulated packets which may be transferred in a contiguous manner on transmit FIFO 160 to the appropriate packet destination (or "next hop").

Header and packet data are transferred in K-bit wide bursts, where K is greater than one. All, half or none of the bits in each burst may be valid. When all bits are valid, a burst transfers a whole width of data. When half of the bits are valid, the burst transfers a half-width of data. Thus, by way of example, where K is sixteen, each burst may transfer two valid bytes (a whole width), one valid byte (a half-width) or no valid bytes. Of course, the value of K may differ in other embodiments without departing from the inventive scope.

Referring now to Figure 2, packet assembly 150 is shown in greater detail. Alignment unit 210 receives stripped packet data in bursts from packet FIFO 130, performs necessary realignments, and forwards the data to update unit 220. Stripped packet data includes inbound packet contents, excluding the stripped inbound header. Update unit 220 separately receives outbound header data in bursts from header FIFO

145. Update unit 220 makes necessary updates to received data and forwards the data in bursts to merger multiplexor 230. Merger multiplexor 230 merges outbound header data and stripped packet data into an encapsulated packet suitable for contiguous transmission on transmit FIFO 160, and delivers the encapsulated packet in bursts to assembly register 240. Updates and merger are assisted by multiplexor control 250. Multiplexor control 250 instructs the multiplexors in update unit 220 and merger multiplexor 230 to select appropriate half-widths from among the currently available widths of data to achieve the desired updates (in the case of update unit 220) and to form the encapsulated packet (in the case of merger multiplexor 230).

A significant advantage of the present invention is "on the fly" selective alignment and merger of separately-sourced outbound header and stripped packet data to form encapsulated packets. The scenarios requiring the application of different alignment and merger rules are shown in Figures 3A through 3D. In Figure 3A, inbound header 310 and outbound header 311 both end on a whole width. Therefore, stripped packet 312 and outbound header 311 both begin on a whole width and may be combined into encapsulated packet 313 without alignment or merger. In Figure 3B, inbound header 320 and outbound header 321 both end in a half-width. Therefore, stripped packet 322 and outbound header 321 both begin on a half-width and may be combined into encapsulated packet 323 without alignment, but require merger in order to form encapsulated packet 323. In Figure 3C, inbound header 330 ends in a whole width while outbound header 331 ends in a half-width. Therefore, stripped packet 332 and outbound header 331 require both alignment and merger in order to form encapsulated packet 333. Finally, in Figure 3D, inbound header 340 ends in a half-width while outbound header 341 ends in a whole

width. Therefore, stripped packet 342 and outbound header 341 require alignment, but do not require merger, in order to form encapsulated packet 343.

Turning now to Figure 4, alignment unit 210 is illustrated in even greater detail. Unit 210 includes half-width register 410 followed by full-width registers 420, 430. The stripped packet arrives at unit 210 from packet FIFO 130 in a series of bursts, which are operatively treated as half-burst pairs. Once at unit 210, bypass logic 440 directs the half-widths in the half-burst pairs on one of three paths, depending on (i) whether or not there is a half-width divergence between the inbound header and the outbound header and (ii) whether, if there is a half-width divergence, the outbound header ends on a half-width or a whole width. In a preferred embodiment, bypass logic 440 makes these two determinations by consulting the least significant bit of an offset field in the stripped packet (which indicates whether the start of the stripped packet was offset from the start of the inbound header by a half-width and, therefore, whether the stripped packet begins on a half-width or a whole width) and the least significant bit in a header length field in the outbound header (which indicates whether the outbound header ends on a half-width or a whole width). All paths are selected to advance the half-widths in the half-burst pairs through the alignment unit 210 on two burst cycles, although paths requiring a different or variable number of burst cycles may be configured without departing from the inventive concept.

Referring now to Figure 5, update unit 220 is shown in greater detail. Update unit 220 receives the stripped packet from alignment unit 210 in a series of bursts, which are operatively treated as half-burst pairs, and receives the outbound header from header FIFO 145 in a series of whole bursts. The half-widths of the stripped packet are inputs to

half-width multiplexors 520, while the whole widths of the outbound header are inputs to whole width multiplexor 530. Multiplexors 520, 530 also have as inputs data from various ones of update registers 510. On each burst cycle, multiplexor control 250 instructs each of multiplexors 520, 530 to select one of the inputs as an output, depending on the result of a comparison of update flags with the current burst count. In this regard, multiplexor control 250 has update flags sufficient to indicate for each of update registers 510 the burst cycles on which the multiplexors are to select as outputs the inputs from the various update registers. A burst counter associated with multiplexor control 250 is incremented on each burst cycle and the current counter value is compared for a match with the update flags. When a match is found, multiplexor control 250 instructs the appropriate one or both of multiplexors 520, 530 to select the input from the update register corresponding to the matching update flag. When no match is found, multiplexors 520, 530 select the input from alignment unit 210 and/or header FIFO 145, respectively. By implementing the foregoing, packet fields which require modification on a packet-by-packet basis can be updated. Fields requiring packet-specific modification may include, by way of example, those which indicate the length of a packet or the length of a packet header and/or those which indicate the number of "hops" a packet has traversed or its remaining time-to-live. Although in a preferred embodiment update unit 220 is shown to precede merger multiplexor 230 in series, it may be interposed after merger multiplexor 230 in other embodiments.

Turning to Figure 6, merger multiplexor 230 and its associated logic is shown in greater detail. Merger multiplexor 230 receives as inputs the stripped packet and outbound header (as modified by update unit 220). The stripped packet arrives in a series

of half-burst pairs and the outbound header arrives in a series of whole bursts which may be operatively treated as half-burst pairs by multiplexor 230. On each burst, multiplexor control 250 instructs merger multiplexor 230 to select as outputs two of the upon to four half-width inputs, depending on the application of a selection matrix to the current burst count. In this regard, multiplexor control 250 has access to merger information sufficient to indicate (i) the total length (in half-widths) of the outbound header; (ii) whether or not there was a half-width divergence between the inbound header and the outbound header; and (iii) whether the outbound header ends on a whole width or a half-width. From the foregoing information, a complete selection matrix for the encapsulated packet is resolved. The burst counter associated with multiplexor control 250 is incremented on each burst cycle and the selection matrix is applied to the current counter value to obtain a selection instruction. The selection instructions are used to control merger multiplexor 230 to select as outputs on the current burst cycle the two half-width inputs which are required in order to successfully merge the separately-sourced outbound header and stripped packet (as modified by update unit 220) into an encapsulated packet arranged for contiguous transmission. The selected whole width outputs are delivered to assembly register 240 for temporary storage en route to transmit FIFO 160.

In Figures 7A through 7D, the selective alignment and merger operation of the present invention is shown for the four possible inbound/outbound header combinations. It is assumed for clarity in Figures 7A through 7D that no half-widths (e.g., A, B, C, D) are displaced in update unit by substitute half-widths (e.g., A', B', C', D') and that the arrival of half-widths at assembly registers is delayed by a fixed number of burst cycles U.

Figure 7A illustrates the scenario wherein the inbound header and outbound header do not have a half-width divergence and the outbound header ends on a whole width. .In that situation, the stripped packet is arranged in packet FIFO 701A to begin on a whole width A/B while the outbound header is arranged in header FIFO 711A to end on a whole width Y/Z. On burst cycle N, initial whole width A/B is stored in packet FIFO 701A as shown. Because there is no half-width divergence, realignment of the stripped packet is not required. Therefore, on burst cycle N+1, whole width A/B bypasses half-width register 702A and flows into full-width register 703A. Separately, on burst cycle N+1, terminal whole width Y/Z of the outbound header is aligned for arrival from header FIFO 711A as shown. On burst cycle N+2, initial whole width A/B of the stripped packet advances to full-width register 704A. On burst cycle N+2+U, terminal whole width Y/Z of the outbound header is selected by merger multiplexor 721A and delivered into assembly register. On burst cycle N+3+U, initial whole width A/B of the stripped packet is selected by merger multiplexor 721A and delivered into assembly register 722A.

Figure 7B illustrates the scenario wherein .he inbound header and outbound header do not have a half-width divergence but the outbound header ends on a half-width. In that situation, the stripped packet is arranged in packet FIFO 701B to begin on a half-width A while the outbound header ends on a half-width Z. On burst cycle N, initial half-width A and whole width B/C which immediately follows are queued in packet FIFO 701B as shown. Because there is no half-width divergence, realignment of the stripped packet is not required. Therefore, on burst cycle N+1, half-width A bypasses half-width register 702B and flows into full-width register 703B. On burst cycle N+2, half-width A

advances to full-width register 704B while whole width B/C bypasses half-width register 702B and flows into full-width register 703B. Separately, on burst cycle N+2, terminal half-width Z of the outbound header is aligned for arrival from header FIFO 711B as shown. On burst cycle N+3, whole width B/C of the stripped packet advances to full-width register 704B. On burst cycle N+3+U, terminal half-width Z of the outbound header and initial half-width A of the stripped packet are selected by merger multiplexor 721B and delivered into assembly register 722B. On burst cycle N+4+U, whole width B/C of the stripped packet is selected by merger multiplexor 721B and delivered into assembly register 722B.

Figure 7C illustrates the scenario wherein the inbound header and outbound header have a half-width divergence and the outbound header ends on a half-width. In that situation, the stripped packet is arranged in interim FIFO 701C to begin on a whole width A/B while the outbound header ends on a half-width Z. On burst cycle N, initial whole width A/B and whole width C/D which immediately follows are queued in packet FIFO 701C as shown. Because there is a half-width divergence and the outbound header ends on a half-width, a whole width to half-width realignment of the stripped packet is required. Therefore, on burst cycle N+1, half-width B is stored in half-width register 702C while half-width A flows into full-width register 703C. On burst cycle N+2, half-width A advances to full-width register 704C while realigned whole width B/C flows into full-width register 703B. Separately, on burst cycle N+2, terminal half-width Z of the outbound header is aligned for arrival from header FIFO 711C as shown. On burst cycle N+3, whole width B/C of the stripped packet advances to full-width register 704C. On burst cycle N+3+U, terminal half-width Z of the outbound header and initial half-width A

of the packet base are selected by merger multiplexor 721C and delivered into assembly register 722C. On burst cycle N+4+U, whole width B/C of the stripped packet is selected by merger multiplexor 721C and delivered into assembly register 722C.

Finally, Figure 7D illustrates the scenario wherein the inbound header and outbound header have a half-width divergence and the outbound header ends on a whole width. In that situation, the stripped packet is arranged in packet FIFO 701D to begin on a half-width A while the outbound header ends on a whole width Y/Z. On burst cycle N, initial half-width and whole width B/C which immediately follows are queued in packet FIFO 701C as shown. Because there is a half-width divergence and the outbound header ends on a whole width, a half-width to whole width realignment of the stripped packet is required. Therefore, on burst cycle N+1, half-width A is stored in half-width register 702D. Separately, on burst cycle N+1, terminal whole width Y/Z of the outbound header is aligned for arrival from header FIFO 711D as shown. On burst cycle N+2, realigned whole width A/B advances to full-width register 704D and half-width C flows into half-width register 702D. On burst cycle N+2+U, terminal whole width Y/Z of the outbound header and is selected by merger multiplexor 721D and delivered into assembly register 722D. On burst cycle N+3+U, whole width A/B of the packet base is selected by merger multiplexor 721D and delivered into assembly register 722D.

It will be appreciated by those of ordinary skill in the art that by performing the foregoing alignment and merger operations, outbound headers and corresponding stripped packets are combined into encapsulated packets in a way which advantageously allows them to be transmitted in a contiguous fashion to their resolved destinations. It will also be appreciated that the invention can be embodied in other specific forms

without departing from the spirit or essential character hereof. There present invention is therefore considered in all respects illustrative and not restrictive. The scope of the invention is indicated by the appended claims, and all changes that come within the meaning and range of equivalents thereof are intended to be embraced therein.

I claim:

1. A method for combining a first and second logical block of data received from separate inputs into a contiguous logical block of data for delivery to a shared output, wherein the first and second blocks are input in a succession of half-widths at a rate of up to one width per burst cycle, and wherein the first and second blocks begin and end on either half-widths or whole widths, comprising:

(a) determining whether there is a half-width divergence between the first block and the second block;

(b) if there is a half-width divergence, realigning each input of the second block by a half-width;

(c) on each burst cycle before the last half-width of the first block is available for delivery, delivering a width of the first block to the output;

(d) on the burst cycle on which the last half-width of the first block is available for delivery, if the first block ends on a half-width, delivering a half-width of the first block and a half-width of the second block to the output, and otherwise delivering a width of the first block to the output;

(e) on each burst cycle thereafter before the last half-width of the second block is available for delivery, delivering a width of the second block to the output; and

(f) on the burst cycle on which the last half-width of the second block is available for delivery, if the second block as realigned in step (b), if at all, ends on a half-width, delivering a half-width of the second block to the output, and otherwise delivering a width of the second block to the output.

2. The method according to claim 1, further comprising:

(g) on a selected burst cycle, replacing a half-width of the first or second block with a substitute half-width.

3. The method according to claim 1, further comprising:

(g) on a selected burst cycle, replacing a width of the first or second block with a substitute width.

4. A method for combining a first and second logical block of data received from separate inputs into a contiguous logical block of data for delivery to a shared output, wherein the first and second blocks are input and begin and end on either half-widths or whole widths, comprising:

(a) aligning the first block and the second block to the same half-width;

(b) delivering to the output first block inputs until the current first block input for delivery includes the last half-width of the first block;

(c) when the current first block input for delivery includes the last half-width of the first block, if the current first block input for delivery is a half-width, delivering to the output the current first block input for delivery and the current second block input for delivery, and otherwise delivering to the output the current first block input for delivery; and

(d) thereafter, delivering to the output second block inputs until the entire second block has been delivered.

5. The method according to claim 4, further comprising:

(e) replacing a selected half-width of a first or second block input with a substitute half-width.

6. The method according to claim 4, further comprising:

(e) replacing a selected first or second block input with a substitute input.

7. A method for combining an outbound header and a stripped packet received from separate inputs into an encapsulated packet for delivery to a shared output, wherein the outbound header and stripped packet are input in a succession of half-widths at a rate of up to one width per burst cycle, and wherein the outbound header ends and the stripped packet begins on either a half-width or a whole width, comprising:

(a) determining whether an inbound header stripped from the inbound packet and an outbound header have a half-width divergence;

(b) if there is a half-width divergence between the inbound header and the outbound header, realigning each input of the stripped packet by a half-width;

(c) on each burst cycle before the last half-width of the outbound header is available for delivery, delivering a width of the outbound header to the output;

(d) on the burst cycle on which the last half-width of the outbound header is available for delivery, if the outbound header ends on a half-width, delivering the last half-width of the outbound header and the first half-width of the stripped packet to the output, and otherwise delivering the last width of the outbound header to the output;

(e) on each burst cycle thereafter before the last half-width of the stripped packet is available for delivery, delivering a width of the stripped packet to the output; and

(f) on the burst cycle on which the last half-width of the stripped packet is available for delivery, if the stripped packet as realigned in step (b), if at all,

ends on a half-width, delivering a half-width of the stripped packet to the output, and otherwise delivering a width of the stripped packet to the output.

8. The method according to claim 7, further comprising:

(g) on a selected burst cycle, replacing a selected half-width of the outbound header or stripped packet with a substitute half-width.

9. The method according to claim 7, further comprising:

(g) on a selected burst cycle, replacing a selected width of the outbound header or stripped packet with a substitute width.

10. A system for combining a first and second block of data received from separate inputs into a contiguous block of data for delivery to a shared output, wherein the first and second blocks are input in a succession of half-widths at a rate of up to one width per burst cycle, and wherein the first block ends and the second block begins on either a half-width or a whole width, comprising:

an alignment unit having a half-width register and a full-width register in series, the alignment unit arranged to receive widths of the second block, to selectively implement the half-width register to store half-widths of the second block, to store widths of the second block in the full-width register and to transmit widths of the second block from the full-width register; and

a merger multiplexor following the alignment unit in series, the merger multiplexor arranged to receive widths of the first block and the second block, to select two half-widths from among the received widths and to transmit the selected widths.

11. The system according to claim 10, wherein the half-width register is implemented only when there is a half-width divergence between the first block and the second block.

12. The system according to claim 10, wherein the alignment unit has a second full-width register, the second full-width register being selectively implemented.

13. The system according to claim 12, wherein the second full-width register is implemented unless the first block ends on a whole width and the second block begins on a half-width.

14. The system according to claim 10, further comprising:

an update unit arranged to receive widths and substitute half-widths, to select widths from among the received widths and half-widths and to transmit the selected widths.

15. The system according to claim 10, further comprising:

an update unit following the alignment unit and preceding the merger multiplexor in series, the update unit arranged to receive widths of the first block and second block and substitute half-widths for the first block and second block, to select widths for each of the first block and second block from among the received widths and half-widths and to transmit the selected widths of the first and second blocks to the merger multiplexor.

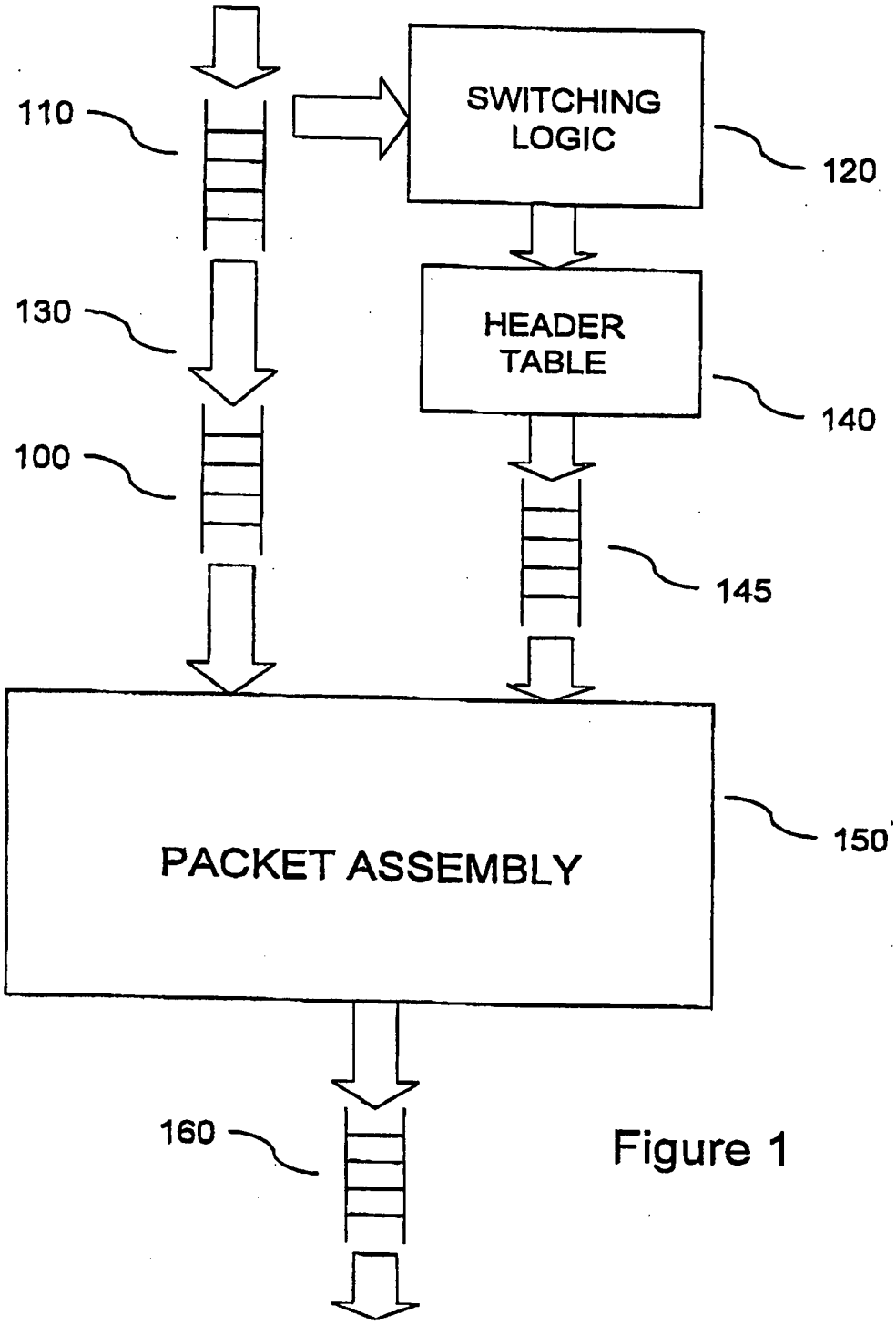
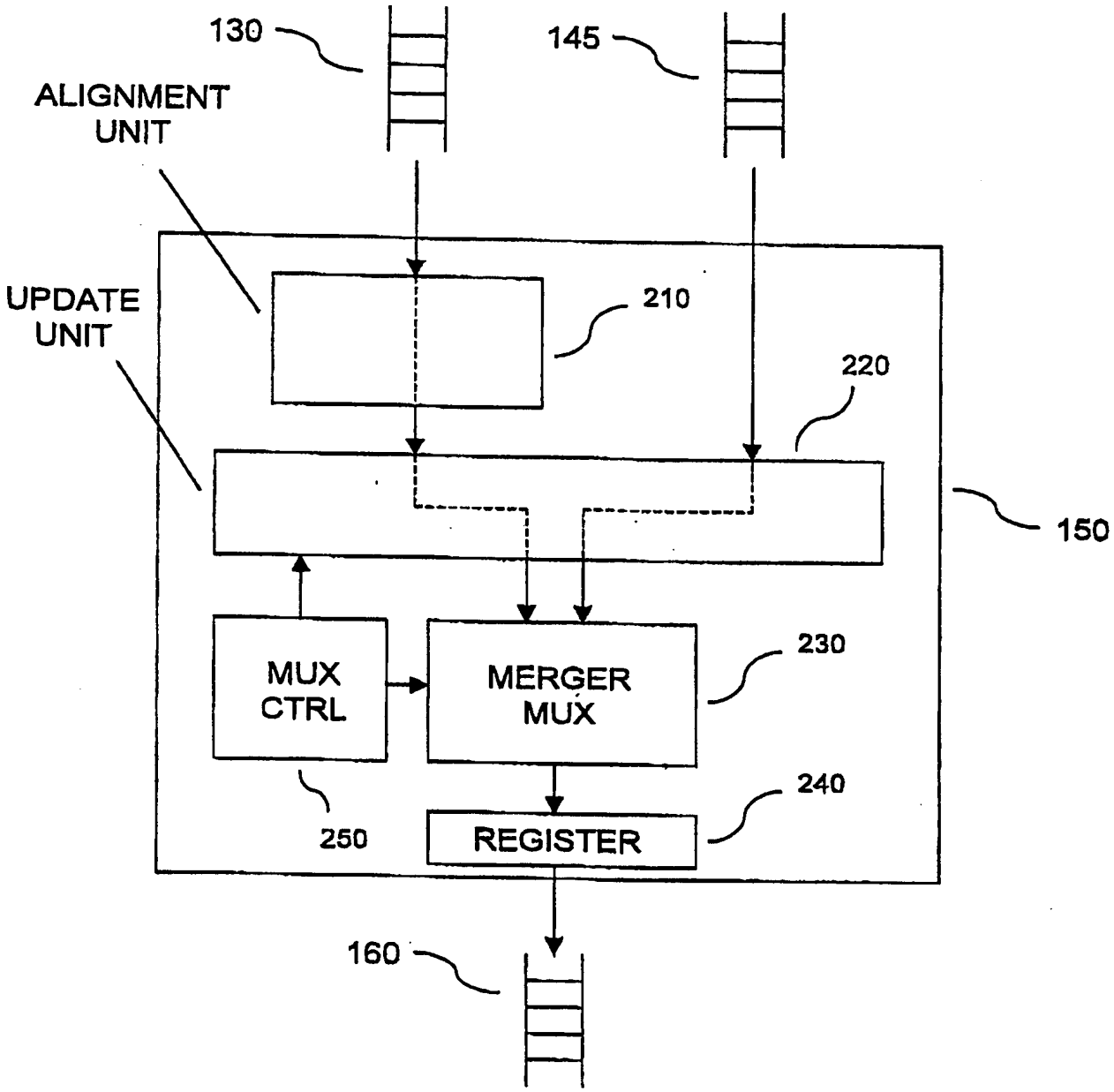

110
SWITCHING LOGIC
120
130
HEADER TABLE
140
100
145
PACKET ASSEMBLY
150
160

Figure 1

Figure 2


130
145
ALIGNMENT UNIT
UPDATE UNIT
210
220
150
MUX CTRL
MERGER MUX
230
240
250
REGISTER
160

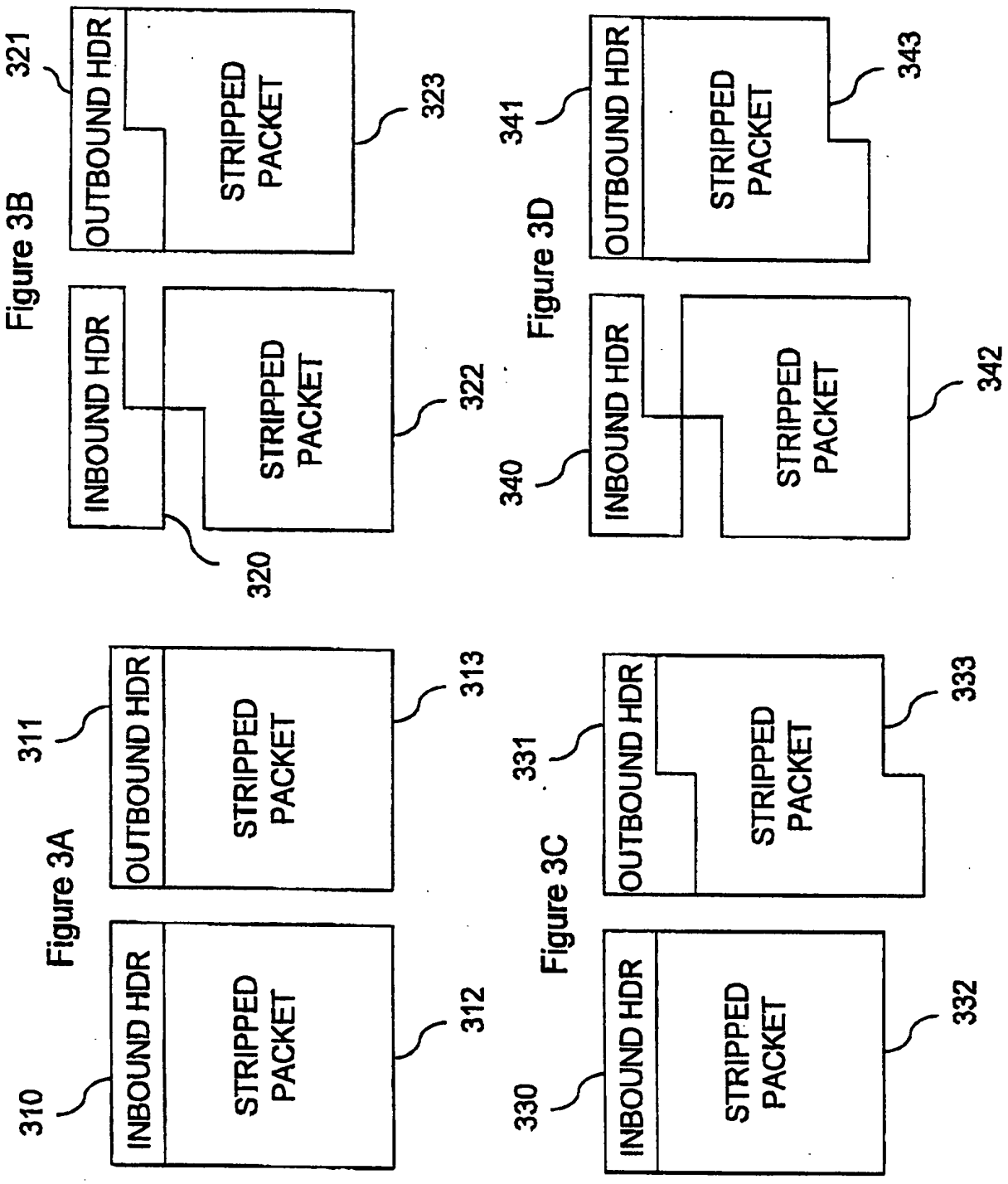


Figure 3A
310
INBOUND HDR
STRIPPED PACKET
312
311
OUTBOUND HDR
STRIPPED PACKET
313
Figure 3B
320
INBOUND HDR
STRIPPED PACKET
322
321
OUTBOUND HDR
STRIPPED PACKET
323
Figure 3C
330
INBOUND HDR
STRIPPED PACKET
332
331
OUTBOUND HDR
STRIPPED PACKET
333
Figure 3D
340
INBOUND HDR
STRIPPED PACKET
342
341
OUTBOUND HDR
STRIPPED PACKET
343

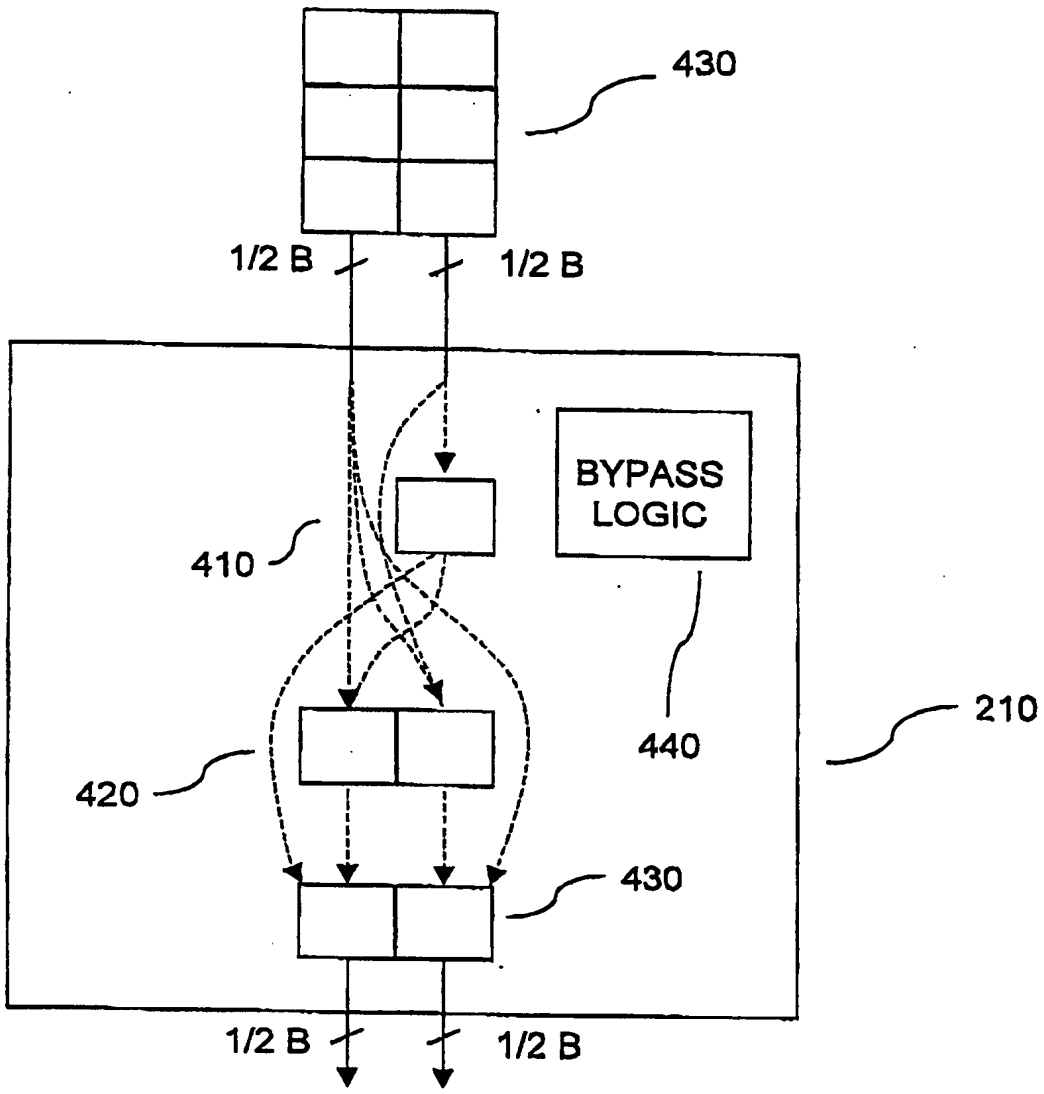


Figure 4

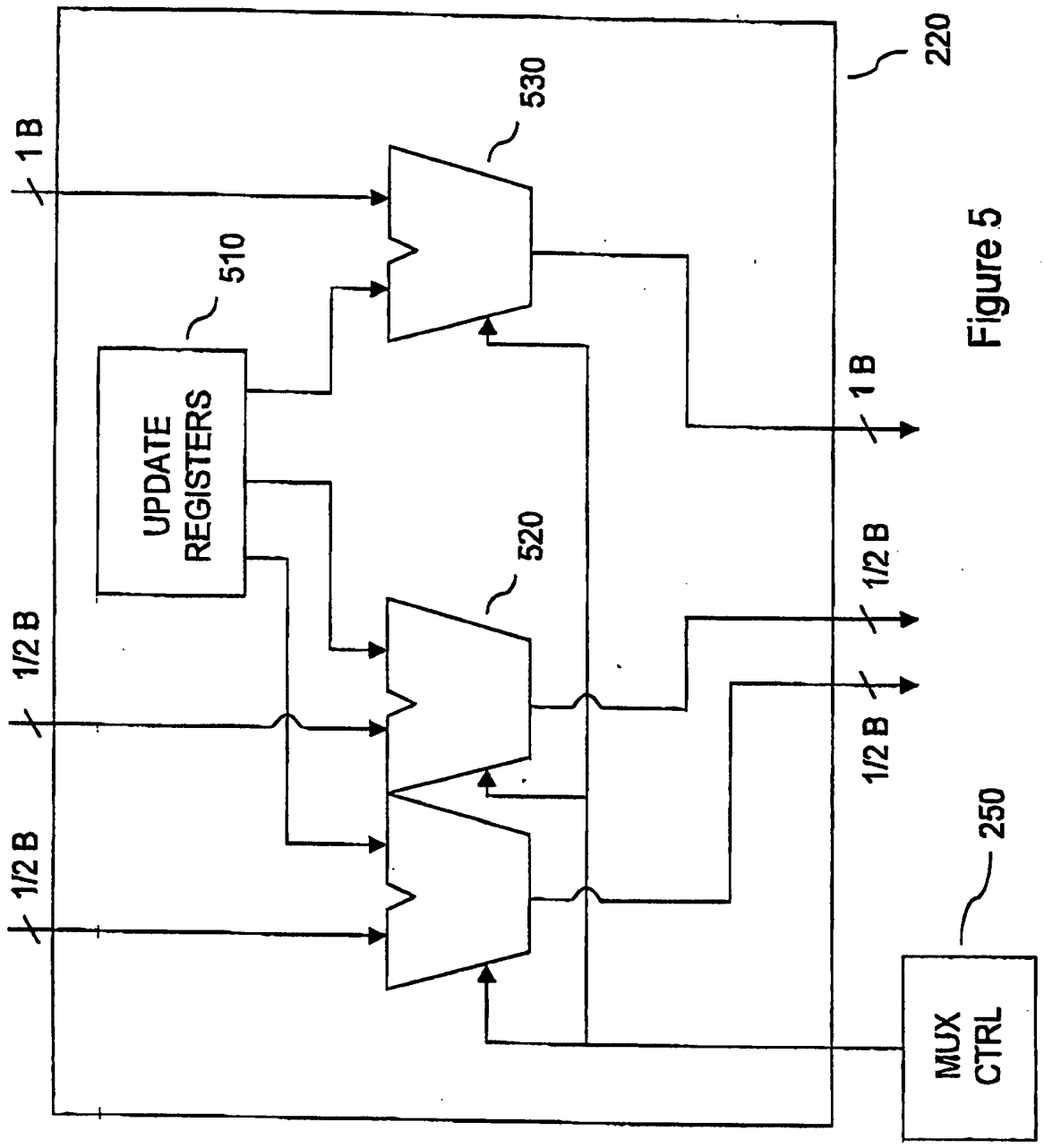
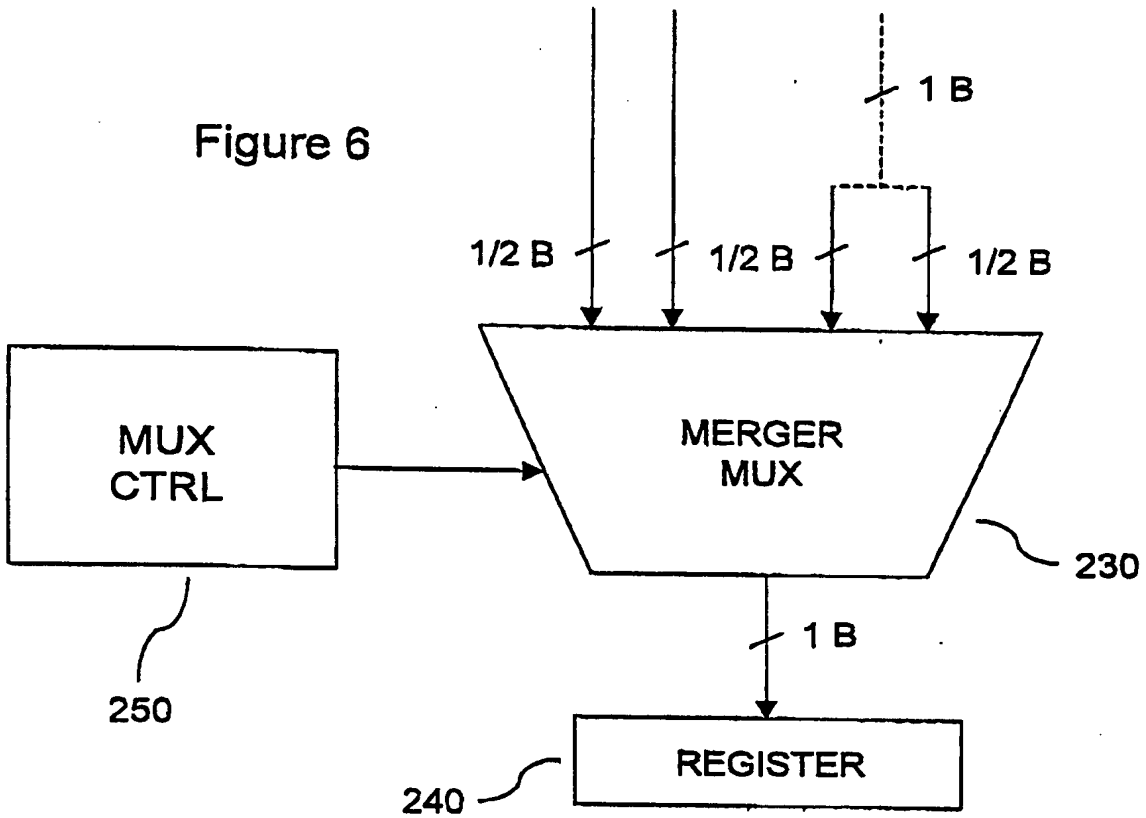

220
530
1 B
510
UPDATE REGISTERS
1 B
520
1/2 B
1/2 B
1/2 B
1/2 B
250
MUX CTRL

Figure 5

Figure 6



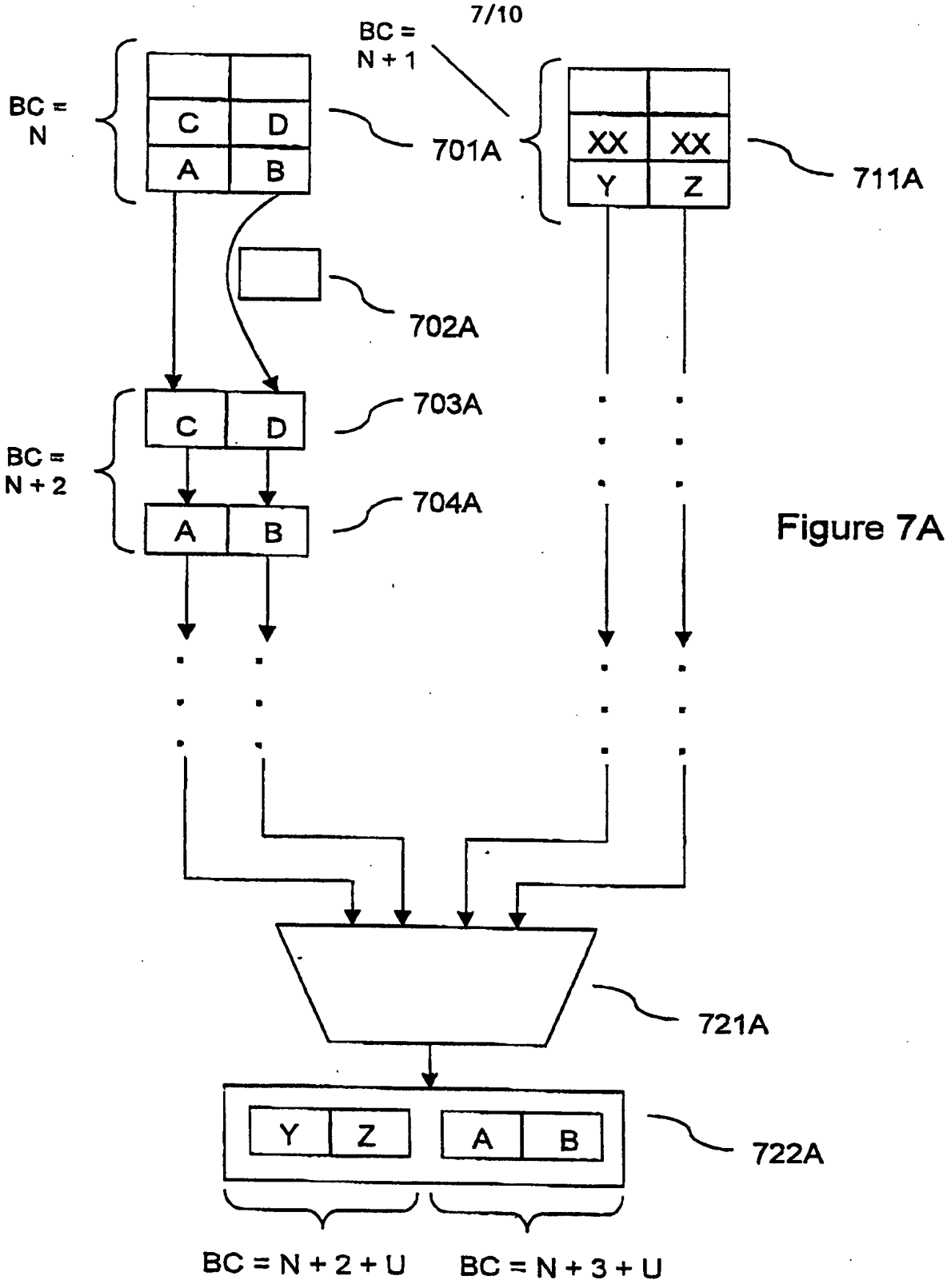

BC = N + 1
BC = N
C
D
A
B
701A
XX
XX
Y
Z
711A
702A
703A
BC = N + 2
704A
721A
Y
Z
A
B
722A
BC = N + 2 + U
BC = N + 3 + U

Figure 7A

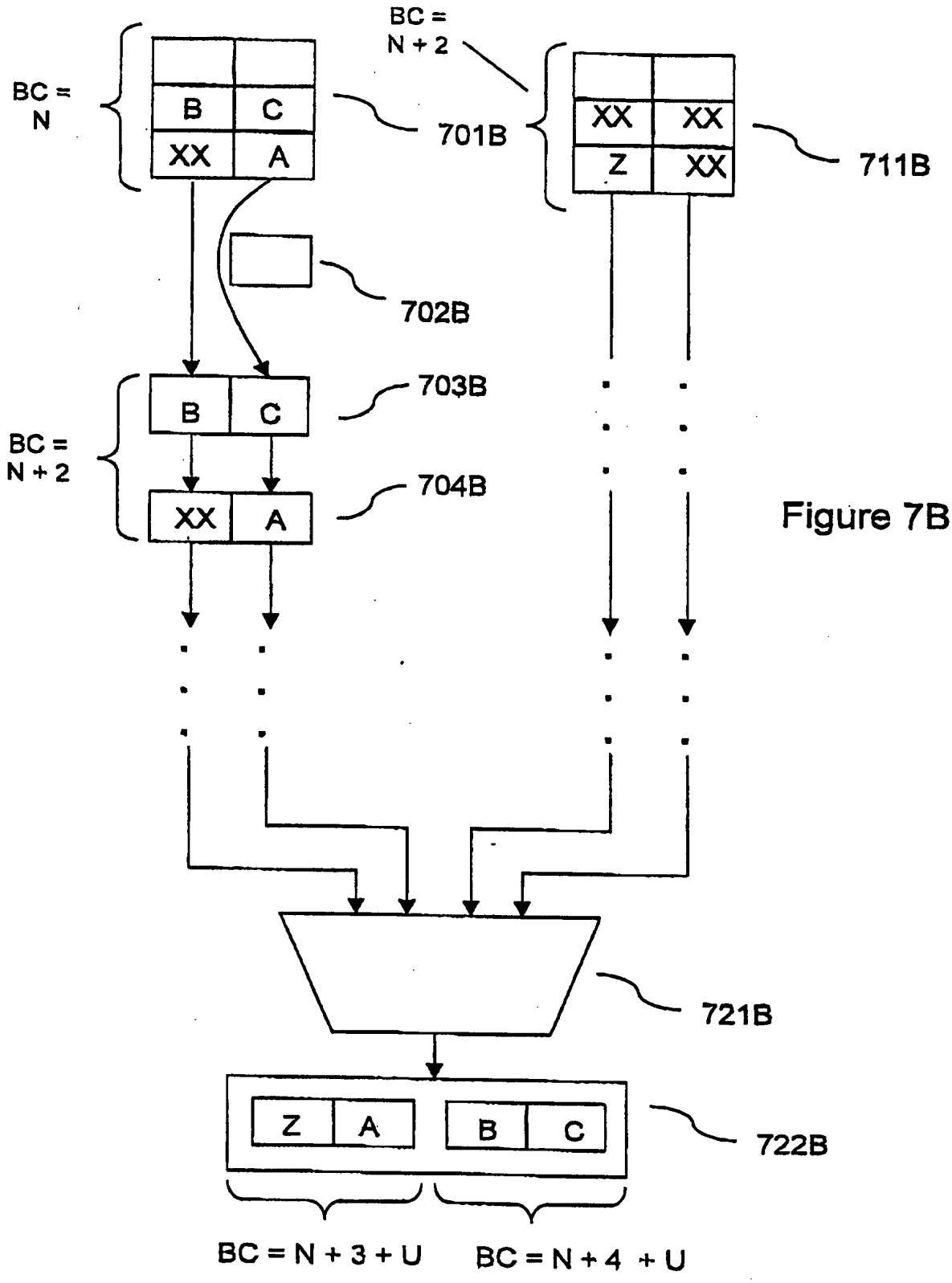

BC = N
B
C
XX
A
701B
BC = N + 2
XX
XX
Z
XX
711B
702B
703B
B
C
BC = N + 2
704B
XX
A
721B
Z
A
B
C
722B
BC = N + 3 + U
BC = N + 4 + U

Figure 7B

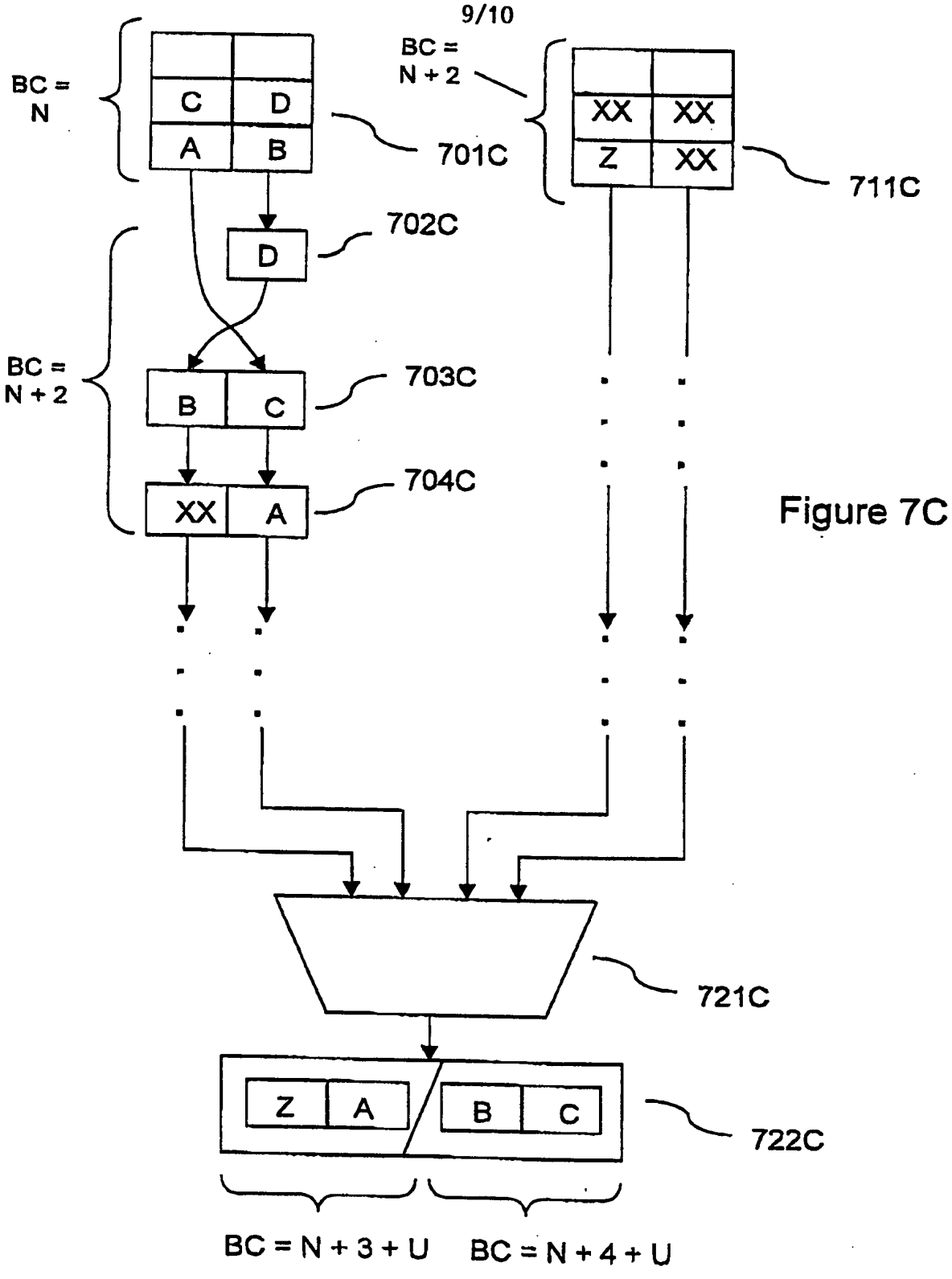

BC = N
C
D
A
B
701C
BC = N + 2
XX
XX
Z
XX
711C
702C
D
BC = N + 2
B
C
703C
XX
A
704C
721C
Z
A
B
C
722C
BC = N + 3 + U
BC = N + 4 + U

Figure 7C

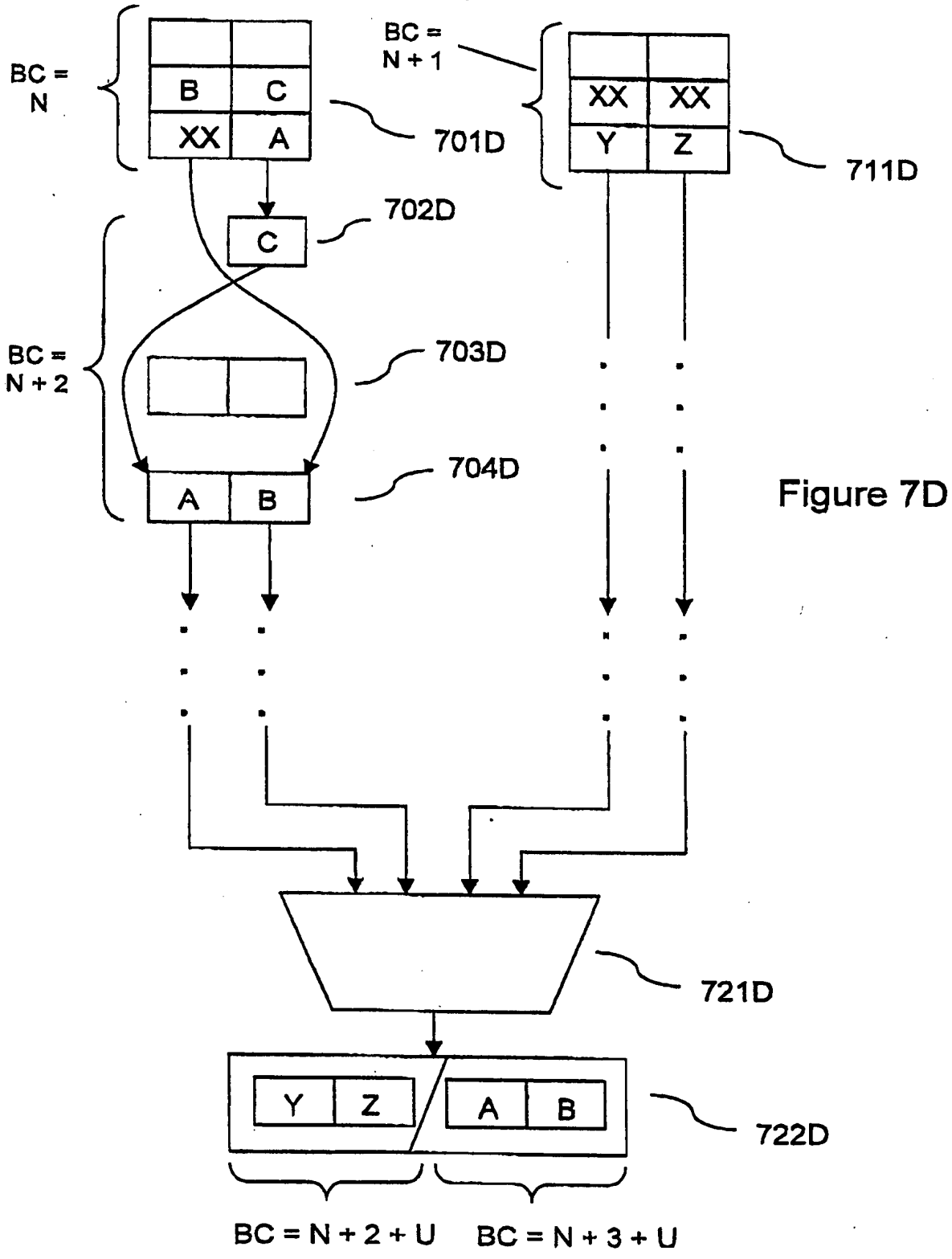

BC = N
B
C
XX
A
701D
BC = N + 1
XX
XX
Y
Z
711D
702D
C
BC = N + 2
703D
704D
A
B
721D
Y
Z
A
B
722D
BC = N + 2 + U
BC = N + 3 + U

Figure 7D

INTERNATIONAL SEARCH REPORT

Inte ·tional Application No
PC /EP 99/04377

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04L12/56

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04L H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	"MECHANISM TO AUTOMATICALLY ALIGN PACKETS IN SWITCH ADAPTERS" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 38, no. 4, 1 April 1995 (1995-04-01), page 279/280 XP000516150 ISSN: 0018-8689 the whole document	1,4,7,10
A	US 5 491 802 A (THOMPSON MICHAEL I ET AL) 13 February 1996 (1996-02-13) column 3, line 11 -column 5, line 28	1,4,7,10

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
27 September 1999	06/10/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Ströbeck, A

1

INTERNATIONAL SEARCH REPORT

.nformation on patent family members

Inte tional Application No
PCI/EP 99/04377

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5491802 A	13-02-1996	DE 69320694 D	08-10-1998
		DE 69320694 T	21-01-1999
		EP 0572145 A	01-12-1993
		JP 6062075 A	04-03-1994